```
1 <?php
 2 /**
 3 * @author GOLAY Brian
 4 * @version 1.0 (2021/05/20)
 5 * Database connection functions
 6 */
 7 namespace FlightClub\sql;
 8 require once 'dbInfos.php';
9
10 //we check if the user session is already existing. If not we initialize it
11 if (!isset($_SESSION['user'])) {
12
       $_SESSION['user'] = "";
13 }
14
15
16
17 class DBConnection {
       static $conn = null; //database connection
18
19
       const NOM BASE = DB NAME;
20
       const HOST = DB HOST;
       const USER = DB USER;
21
       const PWD = DB_PWD;
22
23
24
25
       * Connection to the database function
26
        * @return object return a non persistant connection object if the db connection
27
  was successful, else it return null
28
29
       private static function doConnection() {
30
           try {
31
               self::$conn = new \PDO(
                       "mysql:host=" . self::HOST .
32
                       ";dbname=" . self::NOM_BASE, self::USER, self::PWD,
33
   array(\PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8",
                   \PDO::ATTR_PERSISTENT => false));
34
               self::$conn->setAttribute(\PDO::ATTR_ERRMODE, \PDO::ERRMODE_EXCEPTION);
35
36
           } catch (\Exception $e) {
               echo 'Erreur : ' . $e->getMessage() . '';
37
               die('Could not connect to MySQL');
38
39
40
           return self::$conn;
41
       }
42
43
44
        * create a new instance of a DB connection if not already existing
45
        * @return object
46
47
        */
48
       public static function getConnection() {
           if (self::$conn == null) {
49
50
               self:: doConnection();
51
52
           return self::$conn;
53
       }
54
55
       //TRANSACTION
56
57
        * Begin a transaction
58
```

localhost:4649/?mode=php 1/2

```
20/05/2021
                                                 dbConnection.php
 59
         * @return void
 60
        public static function startTransaction()
 61
 62
        {
             DBConnection::getConnection()->beginTransaction();
 63
         }
 64
 65
         /**
 66
         * Rollback a transaction
 67
 68
         * @return void
 69
 70
 71
        public static function rollback()
 72
        {
 73
             try {
                 DBConnection::getConnection()->rollback();
 74
 75
             } catch (\Throwable $th) {
 76
                 //throw $th;
 77
             }
 78
         }
 79
        /**
 80
         * Commit a transaction
 81
 82
 83
         * @return void
 84
 85
        public static function commit()
 86
         {
 87
             try {
                 DBConnection::getConnection()->commit();
 88
             } catch (\Throwable $th) {
 89
 90
                 //throw $th;
 91
             }
 92
 93
         }
 94
```

95

?>

localhost:4649/?mode=php 2/2